08/08/25

NeuroHackademy 2025

Project Presentation

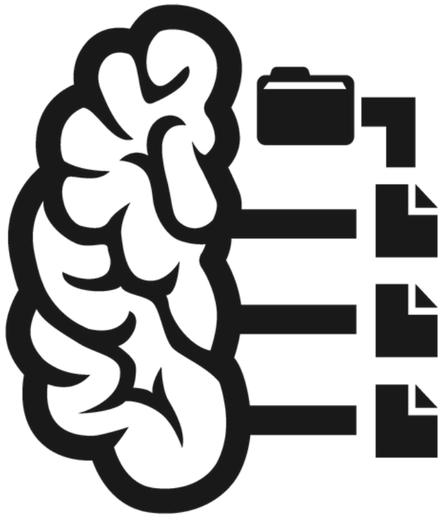
Niv Cohen

Table of Contents

Opening & Motivation

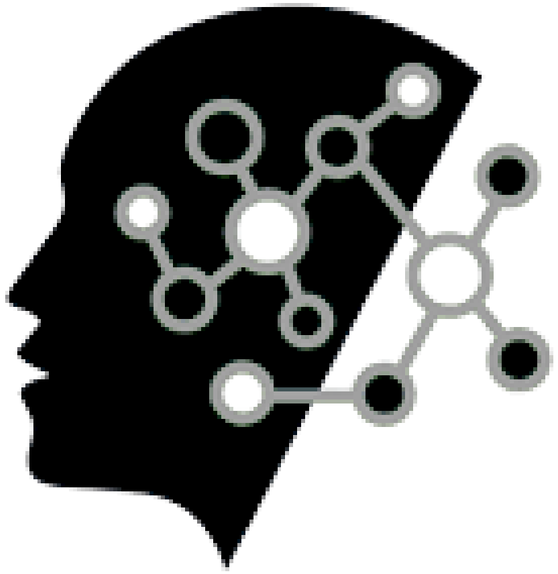
Why I Built bcivis

- Needed a plug-and-play EEG pipeline during my thesis
- Most tools were either too rigid or too open-ended
- Wanted a simple workflow: BIDS data → config file → run



BIDS

BRAIN IMAGING DATA STRUCTURE



BCI

Brain Computer Interface

BIDCI

- BIDCI: Modular EEG preprocessing and visualization for BCI tasks. Built on MNE, structured by BIDS.
- Plug-and-play. Scriptable. Transparent.

General Pipeline

Code

- Load the data
- Preprocessing
- Plot

User

- Install BIDCI
- BIDS format
- Configuration file

Snipped code

The main - test.py

```
import yaml
from src.bcivis.manager.core import DatasetManager


config_path = "C:/Users/ncohe/Desktop/bci_vis/bci-vis-main/src/bcivis/config/motor_imagery_ds003810.yaml"
with open(config_path, "r") as f:
    config = yaml.safe_load(f)

manager = DatasetManager(config=config)
manager.load_all()
manager.preprocess_all()
manager.summarize_all(with_plots=config.get("sanity_check", {}).get("enable_plots", False))
```

Load & preprocess & plot - manager.py

```python
class DatasetManager:
    def __init__(self, config):
        self.config = config
        self.loaders = []
        self.epochs_list = []


    def load_all(self):
        subjects = self.config.get("subjects") or [self.config.get("subject")]
        runs = self.config.get("runs") or [self.config.get("run")]

        if not subjects or not runs:
            raise ValueError("Subjects and runs must be specified in the configuration.")

        for subject in subjects:
            for run in runs:
                loader = BIDSDataLoader(
                    bids_root=self.config["bids_root"],
                    subject=subject,
                    task=self.config["task"],
                    run=run,
                    config=self.config,
```

Load & preprocess & plot - manager.py

```python
def preprocess_all(self):…

def summarize_all(self, with_plots=False):
    for i, loader in enumerate(self.loaders):
        print(f"\n✅ Loader {i+1}")
        print(f"Subject: {loader.subject} | Run: {loader.run}")
        print("Raw info:", loader.raw.info)
        print("Event IDs:", loader.event_id)
        print("Events shape:", loader.events.shape)
        print("Epochs shape:", self.epochs_list[i].get_data().shape)

        subject = loader.subject
        run = loader.run

        if with_plots:
            print("Generating plots...")
            raw = loader.get_raw()
            apply_montage(raw, self.config)
            plot_sensors(raw, self.config, subject=subject, run=run)
            plot_raw(raw, self.config, subject=subject, run=run)
            plot_psd(raw, self.config, subject=subject, run=run)

            event_labels = list(loader.event_id.keys())
            plot_all_conditionwise(self.epochs_list[i], event_labels, self.config, subject=subject, run=run)

            print("Plots generated.")
```

Configuration file - config.yaml

src > bcivis > config > ! motor_imagery_ds003810.yaml

```yaml
bids_root: C:/Users/ncohe/Desktop/bci_vis/BIDS_dataset/motor_imagery_ds003810
task: MIvsRest
subjects: ["02", "04"]
runs: ["1","2"]

class_map: ···

event_id: ···

preprocessing: ···

visualization: ···

sanity_check: ···

output: ···

save_figures: ···
```

Package Structure

V Package strcuture

```
bidci/
│
├── config/                # Dataset-specific configuration
│
├── io/                    # Data loading and BIDS handling
│   └── loader.py
│
├── manager/               # Orchestration and pipeline control
│   └── core.py
│
├── preprocessing/         # Filtering, artifact removal, etc.
│   └── cleaning_pipeline.py
│
├── tasks/                 # Task-specific logic (e.g., motor imagery)
│   └── motor_imagery.py
│
├── utils/                 # Helpers and shared functions
│   └── helpers.py
│
├── vis/                   # Visualization tools
│   └── visualization.py
│
├── test.py                # Sanity check runner
└── README.md              # Project info
```

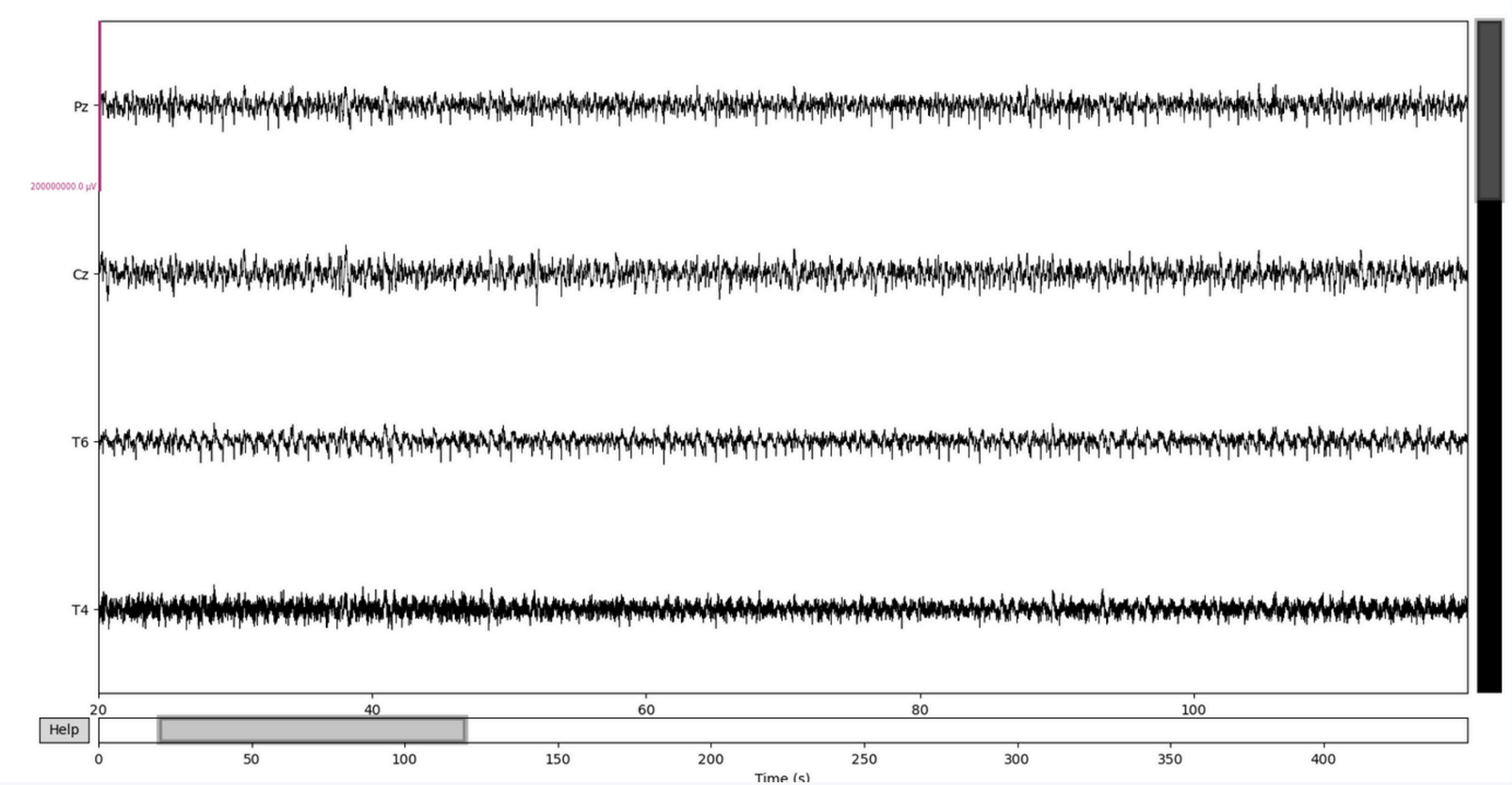
Live Demo

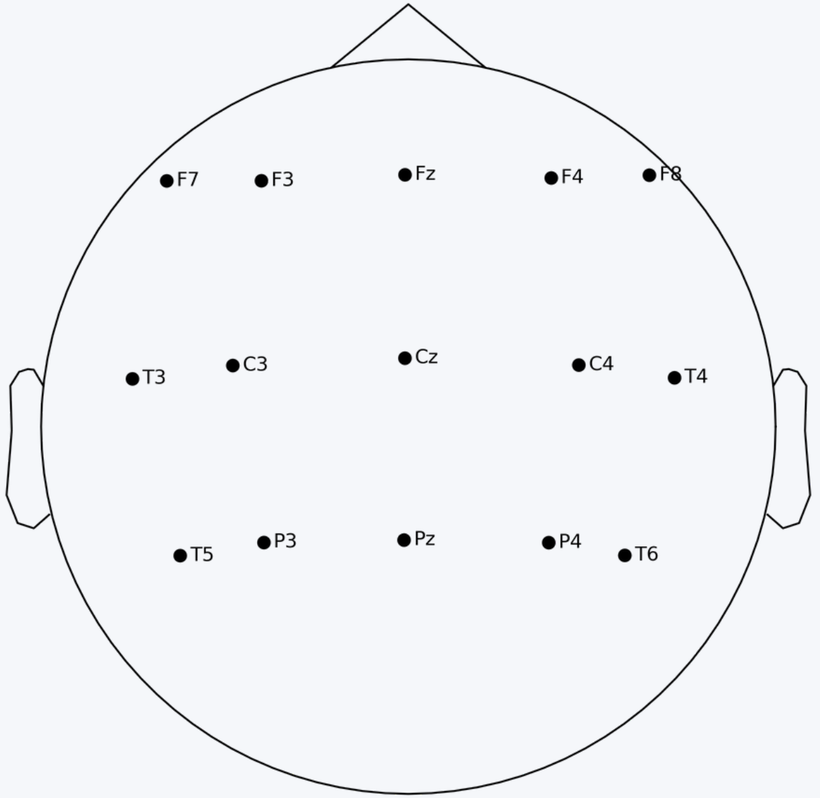
Sumarry

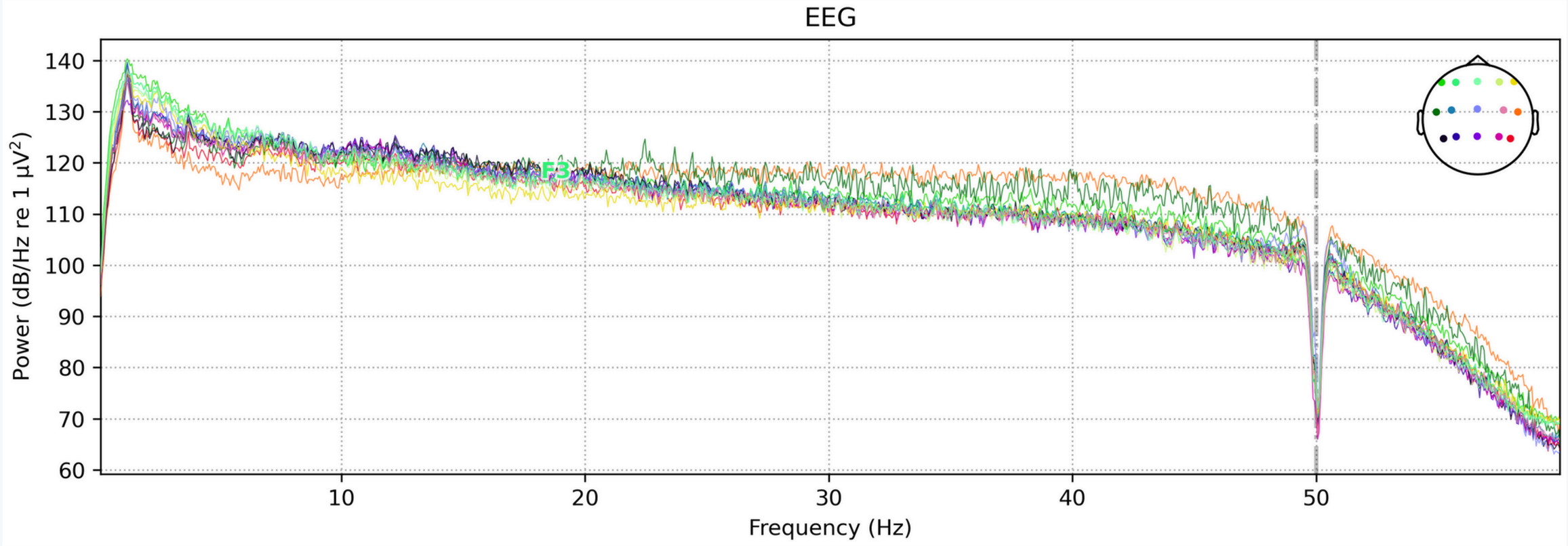
- Introduced BIDCI – a modular, BIDS-compatible EEG pipeline
- Designed for plug-and-play use, focusing on BCI tasks
- Minimal setup: install → BIDS → config → run
- Open-source, extensible, and ready to use

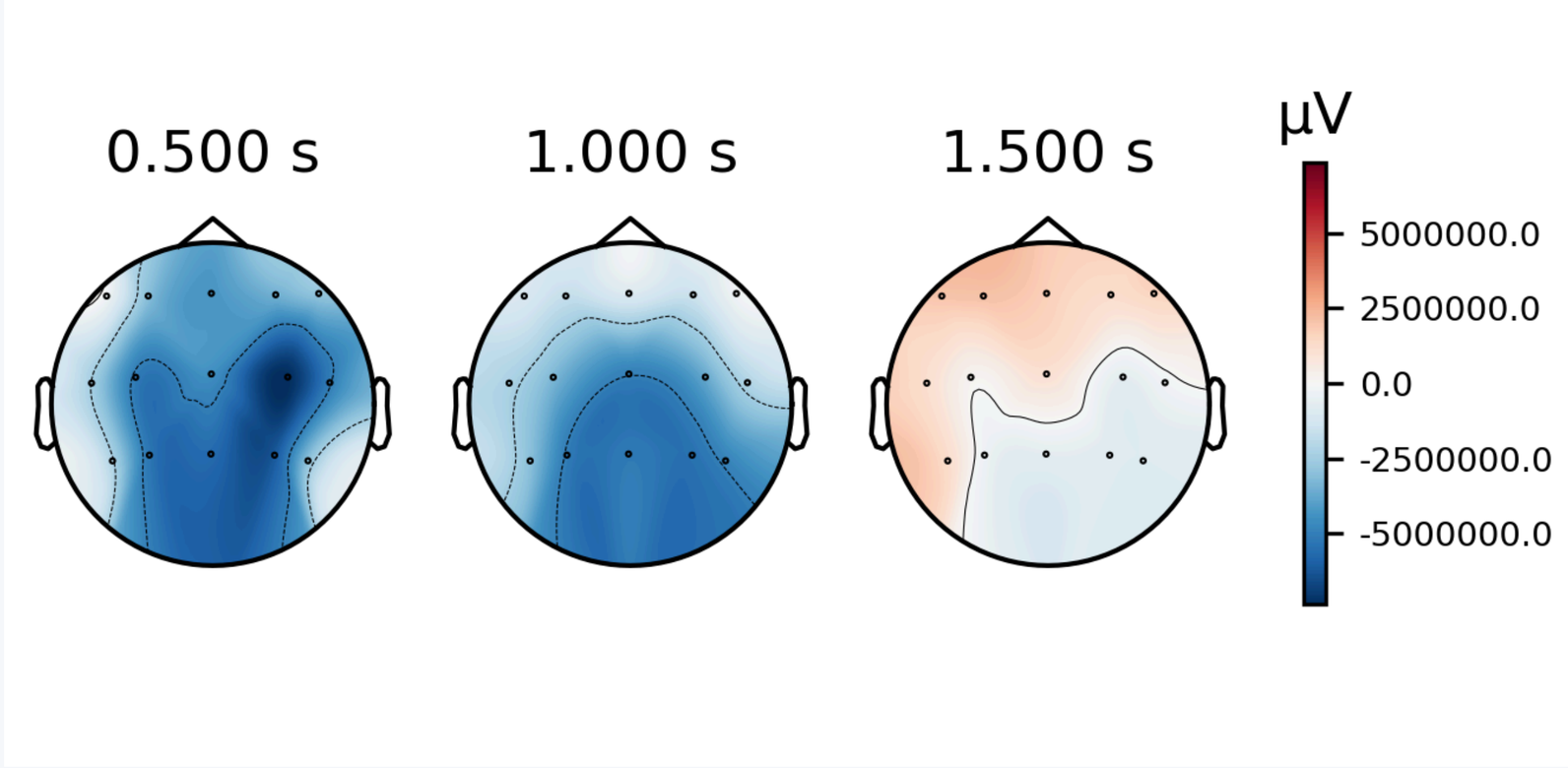
Thank you for listening

Figures


Pz
200000000.0 µV
Cz
T6
T4
20
40
60
80
100
Help
0
50
100
150
200
250
300
350
400
Time (s)


F7
F3
Fz
F4
F8
T3
C3
Cz
C4
T4
T5
P3
Pz
P4
T6


EEG
F3
Power (dB/Hz re 1 µV²)
Frequency (Hz)
140
130
120
110
100
90
80
70
60
10
20
30
40
50


0.500 s
1.000 s
1.500 s
μV
5000000.0
2500000.0
0.0
-2500000.0
-5000000.0